

**Installation de LAMP**

Projet m2l.org

DUMAS Lucie

Table des matières

[LAMP 3](#_Toc145487241)

[Qu’est-ce que LAMP ? 3](#_Toc145487242)

[Le serveur web 4](#_Toc145487243)

[SQLite3 6](#_Toc145487244)

[MariaDB (MySQL) 9](#_Toc145487245)

[PHPMyAdmin 12](#_Toc145487246)

[Finalisation 15](#_Toc145487247)

# LAMP

## Qu’est-ce que LAMP ?

LAMP est un ensemble de logiciels libres permettant de construire des serveurs de sites web. L’acronyme LAMP signifie Linux Apache MySQL (ou MariaDB) PHP :

* « Linux » fait référence au système d’exploitation compatible avec cet ensemble de logiciels. Il en existe également pour Windows (WAMP) ou encore Mac (MAMP).
* Apache est un logiciel de serveurs gratuits et open-source permettant la création et la gestion de serveurs web.
* MySQL est un SGBDR, c’est-à-dire un Système de Gestion de Bases de Données Relationnelles open-source.
* PHP (ou PHP Hypertext Preprocessor) est un langage de scripts généraliste et open-source, spécialement conçu pour le développement d’applications web. Il peut être intégré facilement au HTML.

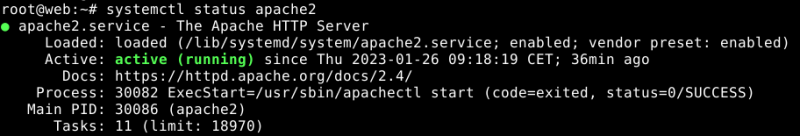
# Le serveur web

Lors de la dernière mission, nous avions installé apache2 afin de créer notre serveur web. Pour plus de détails, nous pouvons nous référer à la mission « Installation de conteneurs LXC ».

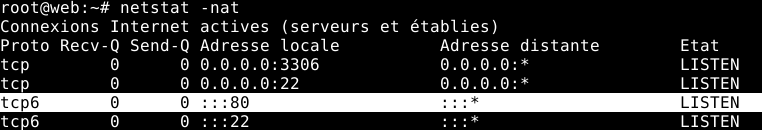
Pour cette mission, nous effectuerons nos manipulations majoritairement dans le conteneur web. Nous pouvons vérifier qu’il soit bien installé et lancé grâce aux commandes netstat et systemctl :

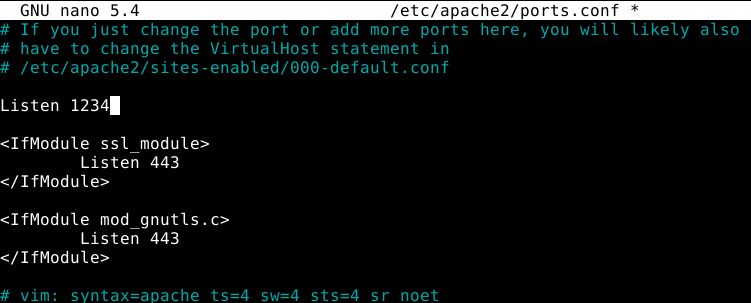
systemctl status apache2

netstat -nat

 La commande systemctl status apache 2 nous permet de voir si le service est en cours d’exécution, s’il est à l’arrêt ou s’il rencontre des difficultés à démarrer.

Nous pouvons également voir lorsque nous effectuons cette commande le PID du service. L’identifiant de processus, ou PID (Process Identifier) est un code unique attribué à tout processus lors de son démarrage (uniquement sur les systèmes Unix ou Windows). Le PID attribué à apache2 lors de ce démarrage est 30086. Le PID est aussi trouvable en effectuant la commande netstat -natp.

La commande netstat -nat nous indique tous les ports en mode « Listen » (ou écoute) de notre conteneur. Pour vérifier qu’apache2 soit fonctionnel, nous vérifions que le port 80 soit bien en écoute :

Pour modifier le port d’écoute d’apache2, nous devons changer le fichier /etc/apache2/ports.conf comme ceci :

Si nous redémarrons le service apache et que nous retapons la commande netstat -nat, nous voyons qu’apache n’écoute plus le port 80. Il écoute maintenant le port 1234.

Pour le service apache2, les fichiers de log sont situés dans le dossier /var/log/apache2

# SQLite3

Nous allons maintenant installer SQLite3. Pour ce faire, nous allons mettre à jour nos paquets et télécharger les nouveaux paquets nécessaires :

apt update && apt upgrade

apt install sqlite3 php-sqlite3 libapache2-mod-php

Voici la liste des modules complémentaires installés :

libapache2-mod-php

libapache2-mod-php7.4

libreadline8

libsodium23

php-common

php-sqlite3

php7.4-cli

php7.4-common

php7.4-json

php7.4-opcache

php7.4-readline

php7.4-sqlite3  
psmisc

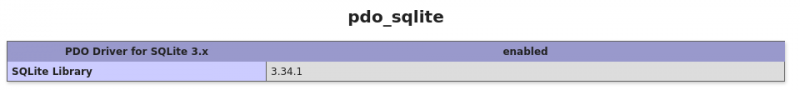
readline-common

sqlite3

Nous créons un script info.php dans notre dossier /var/www/html qui nous servira à vérifier la librairie PDO pour SQLite3 en accédant à notre site : http://10.31.96.50/info.php

<?php

phpinfo() ;

?>

Toujours dans le dossier /var/www/html, nous devons créer une base de données correspondant au cahier des charges à l’aide de la commande suivante et du SQL nécessaire :

sqlite3 myCDS.db

# Création des tables

create table artist (art\_id INTEGER PRIMARY KEY, art\_name TEXT);

create table cd (cd\_id INTEGER PRIMARY KEY, art\_id INTEGER NOT NULL, cd\_title TEXT NOT NULL, cd\_date TEXT);

# Insertion des données

insert into artist (art\_id,art\_name) values (NULL, ‘Peter Gabriel’);

insert into artist (art\_id,art\_name) values (NULL, ‘Bruce Hornsby’);

insert into artist (art\_id,art\_name) values (NULL, ‘Lyle Lovett’);

insert into artist (art\_id,art\_name) values (NULL, ‘Beach Boys’);

insert into cd (cd\_id,art\_id,cd\_title,cd\_date) values (NULL,1,’Us’,’1992’);

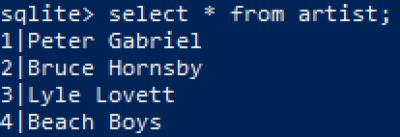
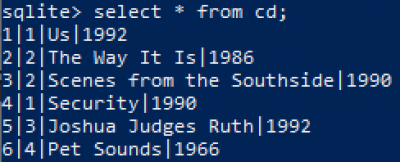
insert into cd (cd\_id,art\_id,cd\_title,cd\_date) values (NULL,2,’The Way It Is’, ‘1986’);

insert into cd (cd\_id,art\_id,cd\_title,cd\_date) values (NULL,2,’Scenes from the Southside’,’1990’);

insert into cd (cd\_id,art\_id,cd\_title,cd\_date) values (NULL,1,’Security’,’1990’);

insert into cd (cd\_id,art\_id,cd\_title,cd\_date) values (NULL,3,’Joshua Judges Ruth’,’1992’);

insert into cd (cd\_id,art\_id,cd\_title,cd\_date) values (NULL,4,’Pet Sounds’, ‘1966’);

 Nous pouvons alors effectuer une requête SQL pour vérifier le contenu de notre base de données :

Nous créons le script PHP bdd.php qui nous permettra d’afficher le contenu de la base :

<?php

try {

$db = new PDO(‘sqlite :myCDS.db’);

print (“<b>Connecté à la base</b><br><br> “) ;

print (“ <b>Les artistes</b><br>“) ;

$sql = ‘SELECT \* from artist’ ;

$result = $db->query($sql) ;

foreach ($result as $row) {

print ($row[‘art\_id’]. “.\t“.$row[‘art\_name’].’<br>’);

};

print (“<br><b>Les albums</b><br>“) ;

$sql = ‘SELECT \* from cd’ ;

$result = $db->query($sql) ;

foreach ($result as $row) {

print ($row['art\_id'].".\t".$row['cd\_title'].".\t".$row['cd\_date'].'<br>');

};

$db = NULL;

}

catch(PDOException $e) {

print (‘Exception : ‘. $e->getMessage());

]

?>

# MariaDB (MySQL)

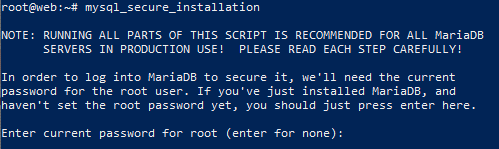
Nous allons maintenant installer MariaDB. Nous mettons à jour nos paquets :

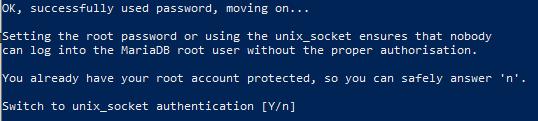
apt update && apt upgrade

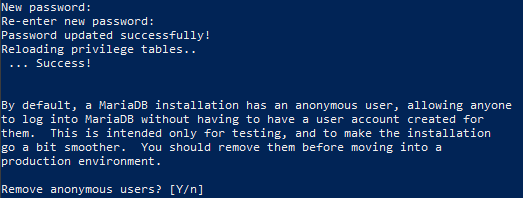
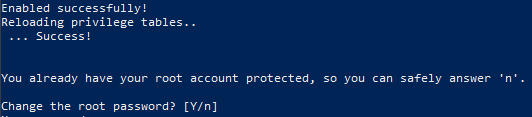
apt install mariadb-server php-mysql

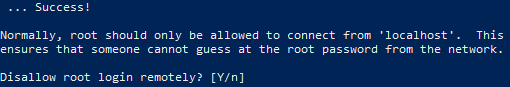
Nous terminons l’installation de MariaDB grâce à l’installation sécurisée. Pour cela, nous entrons la commande :

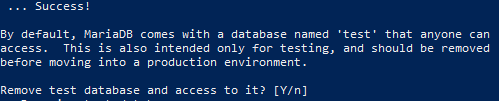
mysql\_secure\_installation

 Dans un premier temps, le script nous demande un mot de passe root (par défaut, il n’en mettra aucun) :

Nous choisissons ensuite toutes les entrées par défaut pour tous les paramètres suivants :







Nous allons maintenant créer un nouvel utilisateur. Pour cela, nous commençons par nous connecter au client MariaDB :

mysql -u root -p

Nous créons ensuite le compte dba qui aura tous les droits sur les bases de données :

# Création de l’utilisateur

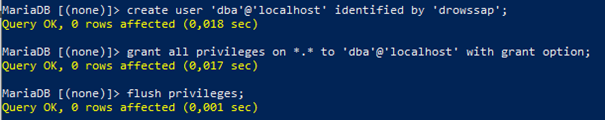
create user ‘dba’@’localhost’ identified by ‘drowssap’;

# Attribution des droits sur toutes les bases de données

grant all privileges on \*.\* to ‘dba’@’localhost’ with grant option;

# Mise à jour des privilèges de la base de données

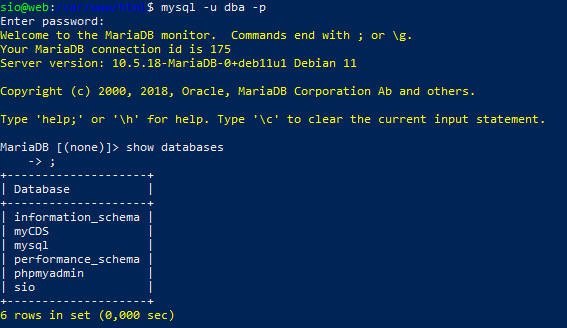
flush privileges ;



Nous pouvons nous déconnecter de la base de données avec Ctrl+D et essayer une nouvelle connexion avec l’utilisateur dba :

mysql -u dba -p

Nous affichons ensuite la liste des bases disponibles :

show databases ;

# PHPMyAdmin

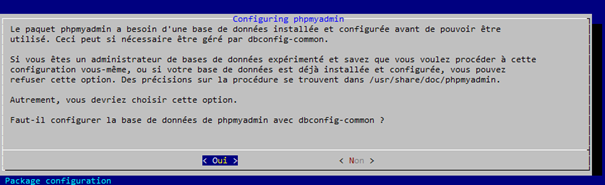
Nous allons maintenant installer PHPMyAdmin. Pour ce faire, nous allons d’abord télécharger les modules PHP nécessaires :

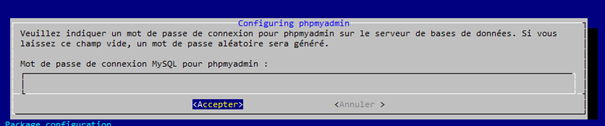
apt update && apt upgrade

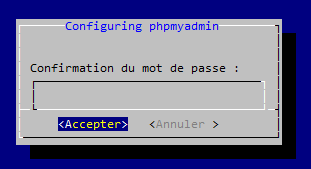
apt install php-json php-mbstring php-zip php-gd php-xml php-curl

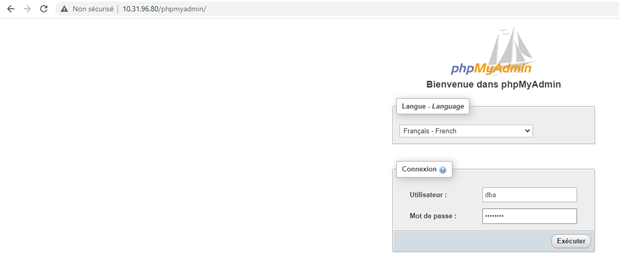
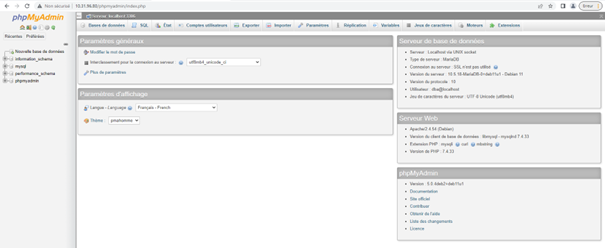
Nous installons ensuite PHPMyAdmin :

apt install phpmyadmin

 Nous suivons les indications en choisissant les mêmes options que sur les captures d’écrans suivantes. Nous commençons par choisir le mode de configuration de PHPMyAdmin par défaut.

 Nous laissons le mots de passe de connexion MySQL par défaut en laissant le champs vide.

 Nous laissons également le champs de confirmation de mot de passe vide.

 Nous nous connectons à PHPMyAdmin via l’interface web à l’adresse 10.31.96.80/phpmyadmin avec l’utilisateur dba créé, afin de tester les manipulations effectuées précédemment.

Enfin, nous entrons les informations de la base de données myCDS.db dans notre nouvelle base de données MariaDB et nous modifions le script bdd.php pour appeler cette nouvelle base de données :

< ?php

try {

$servername = « mysql :host=localhost ;dbname=myCDS”;

$username = “dba”;

$password = “drowssap”;

$db = new PDO($servername, $username, $password);

print (‘<b>Connecté à la base</b><br><br>’) ;

print (‘<b>Les artistes</b><br>’) ;

$sql = ‘SELECT \* from artist’ ;

$result = $db->query($sql) ;

foreach ($result as $row) {

print ($row[‘art\_id’].’.\t’.$row[‘art\_name’].’<br>’)

};

print (‘<br><b>Les albums</b><br>’) ;

$sql = ‘SELECT \* from cd’ ;

$result = $db->query($sql) ;

foreach ($result as $row) {

print ($row[‘art\_id’].’\t’.$row[‘cd\_title’].’.\t’.$row[‘cd\_date’].’<br>’);

};

$db = NULL;

}

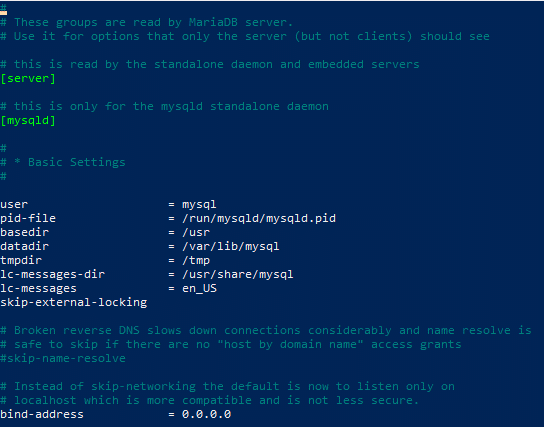
catch(PDOException $e) {

print (‘Exception : ‘. $e->getMessage()):

}

?>

# Finalisation

Par défaut, notre nouvelle base de données écoute sur localhost (l’adresse 127.0.0.1). Nous devons changer cette adresse par défaut pour pouvoir rendre notre base de données accessible à toutes les machines. Pour cela, nous devons modifier le fichier /etc/mysql/mariadb.conf.d/50-server.cnf, plus précisément la ligne bind = 127.0.0.1 que nous devons changer en bind = 0.0.0.0 :

Nous redémarrons le service de MariaDB pour que les changements soient effectifs :

systemctl restart mysql

 Nous voyons maintenant à l’aide de la commande netstat -nat que mysql écoute l’adresse 0.0.0.0, c’est-à-dire toutes les machines.

Nous créons une nouvelle base de données sio pour répondre aux exigences du cahier des charges. Pour cela, nous nous connectons à mariadb à l’aide de la commande suivante et entrons ces commandes successivement :

mysql -u root -p

# Création de la base de données

create database sio ;

# Création de l’utilisateur sio

create user ‘sio’@’localhost’ identified by ‘drowssap’;

# Attribution des privileges (db.privileges to user@localhost)

grant all privileges on sio.\* to ‘sio’@’localhost’ with grant option;

# Mise à jour des privilèges de la base de données

flush privileges ;

